REACT STYLE AND STATE

FULL STACK SKILLS BOOTCAMP

UNDERSTANDING STYLING, ASSETS, AND STATE IN REACT

- **Lesson Overview:**
- In this lesson, we will be introduced to:

1. Methods of applying style in React
2. Using images and other assets
3. React Hooks, useEffect and useState

DIFFERENT WAYS OF APPLYING STYLE IN REACT

- **Inline Styles**:
  Defined directly in JSX elements as an object.
- **Pros**:
  Quick for small, dynamic styles.
- **Cons**:
  Difficult to manage for larger, complex styles.

```
<div style={{ color: "blue", fontSize: "20px" }}>Hello World</div>
```

DIFFERENT WAYS OF APPLYING STYLE IN REACT

- **CSS Stylesheets**:
  Traditional CSS files imported into the component.
- **Pros**:
  Familiar and simple to maintain.
- **Cons**:
  Styles are globally scoped by default.

```
import "./App.css";
```

DIFFERENT WAYS OF APPLYING STYLE IN REACT

- **CSS Modules**:
  CSS files that are locally scoped to components (e.g., App.module.css).
- **Pros**:
  Avoids global scope conflicts.
- **Cons**:
  Requires additional setup and syntax.

```
import styles from "./App.module.css";
<div className={styles.container}>Hello World</div>
```

DIFFERENT WAYS OF APPLYING STYLE IN REACT

- **Styled Components:**
  Styled-components library for writing CSS-in-JS.
- **Pros**:
  Dynamic and component-scoped styling.
- **Cons**:
  Increases bundle size and adds complexity.

```
import styled from "styled-components";
const Button = styled.button`background: blue; color: white;`;
```

Demo…

IMPORTING IMAGES AND ASSETS

- **Importing Images**:
  Import images directly in JavaScript
- **Using public Folder**:
  Place assets in the public folder and reference them
- **When to Use Which**:
  - **src Imports**: Best for component-specific images.
  - **public Folder**: Use for images needed globally across the app.
- **Adding Other Assets**:
  Fonts, icons, and external resources can be imported similarly or linked in index.html.

Demo…

```
import logo from './logo.png';
<img src={logo} alt="Logo" />
```

```
<img src="/logo.png" alt="Logo" />
```

REACT HOOKS: USESTATE

- **What is useState?**
  A React Hook for adding state to functional components.
- **Using useState**:

  useState returns a state variable and a function to update it.

  Allows reactivity, updating the UI when state changes.

```
import { useState } from 'react';
function Counter() {
  const [count, setCount] = useState(0);
  return (
    <div>
      <p>Count: {count}</p>
      <button onClick={() => setCount(count + 1)}>Increment</button>
    </div>
  );
}
```

REACT HOOKS: USEEFFECT

- **What is useEffect?**
  A hook for performing side effects in functional components, such as fetching data, setting up subscriptions, or manually updating the DOM.
- **Using useEffect:**
  The second argument, [], is a dependency array. When empty, the effect runs only once after the initial render.
  Effects can be re-run when dependencies in the array change.

```
import { useEffect, useState } from 'react';
function App() {
  const [data, setData] = useState(null);
  useEffect(() => {
    fetchData().then(response => setData(response));
  }, []);
  return <div>{data ? data : "Loading..."}</div>;
}
```

CONCLUSION

- **Styling Options**:
  - Inline, CSS Stylesheets, CSS Modules, Styled Components.
- **Assets**:
  - How to import images and assets, and when to use src vs. public folder.
- **State and Side Effects**:
  - useState for managing component state.
  - useEffect for side effects and data fetching.

Step8Up Academy

QUESTIONS?